УТВЕРЖДЕН

RU.СГМА.00045-02 13 01-ЛУ

БЕЗОПАСНЫЙ ЛОКОМОТИВНЫЙ ОБЪЕДИНЕННЫЙ КОМПЛЕКС

МАСШТАБИРУЕМЫЙ БЛОК-М

Программное обеспечение

БС-ДПС

Описание программы

RU.СГМА.00045-02 13 01

Листов 26

Инв. № подл.	Подпись и дата	Взам. инв. №	Инв. № дубл.	Подпись и дата

2025

Литера А

АННОТАЦИЯ

В настоящем документе представлены условия применения, назначение и логическая структура программного обеспечения модуля блока согласования с датчиками угла поворота БС-ДПС.

СОДЕРЖАНИЕ

1. ОБЩИЕ СВЕДЕНИЯ

Наименование программы – Программное обеспечение модуля блока согласования с датчиками угла поворота БС-ДПС безопасного локомотивного объединенного комплекса масштабируемого БЛОК-М (далее по тексту – ПО БС-ДПС).

Обозначение программы – RU.СГМА.00045-02.

Язык программирования – AVR Assembler.

ПО БС-ДПС предназначено для работы в составе комплекса БЛОК-М и иных локомотивных систем. Программа записана во внутреннюю память микроконтроллера типа AT90CAN128. Программное окружение для работы ПО БС-ДПС не требуется.

2. ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ

2.1. Основной функциональностью блока ПО БС-ДПС является измерение параметров движения: пройденного пути, скорости и ускорения. Эти величины блок определяет, основываясь на анализе сигналов, приходящих от датчика ДПС.

Дополнительной функциональностью блока является определение местоположения изостыка, основанное на расшифровке кодов КПТ по огибающей АЛСН. Изостыком считается место, в котором произошла смена типа КПТ.

3. ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ

3.1. Состав проекта

3.1.1. Проект включает в себя 316 файлов:

	Имя файла	Назначение файла
1	ArteryAdcDevice.cpp	Драйвер ADC для Artery
2	ArteryAdcDevice.h	Драйвер ADC для Artery
3	ArteryCanDevice.cpp	Драйвер CAN для Artery
4	ArteryCanDevice.h	Драйвер CAN для Artery
5	ArteryCanLoaderIdProvider.cpp	Поставщик идентификационных данных из CAN-загрузчика на Artery
6	ArteryCanLoaderIdProvider.h	Поставщик идентификационных данных из CAN-загрузчика на Artery
7	ArteryFlashMemory.cpp	Внутренняя FLASH-память Artery
8	ArteryFlashMemory.h	Внутренняя FLASH-память Artery
9	ArteryGpioInputDevice.h	Драйвер GPIO, сконфигурированного на вход, для Artery
10	ArteryGpioInterrupt.cpp	Драйвер External interrupt/Event controller для Artery
11	ArteryGpioInterrupt.h	Драйвер External interrupt/Event controller для Artery
12	ArteryGpioOutputDevice.h	Драйвер GPIO, сконфигурированного на выход, для Artery
13	ArteryI2cBus.cpp	Драйвер шины I2C для Artery со стороны Мастера
14	ArteryI2cBus.h	Драйвер шины I2C для Artery со стороны Мастера
15	ArteryRebootService.cpp	Сервис перезагрузки процессора для Artery
16	ArteryRebootService.h	Сервис перезагрузки процессора для Artery
17	ArterySleepService.cpp	Сервис реализации задержки на определённое время для Artery
18	ArterySleepService.h	Сервис реализации задержки на определённое время для Artery
19	ArterySpiSyncMaster.cpp	Драйвер шины SPI со стороны мастера для Artery
20	ArterySpiSyncMaster.h	Драйвер шины SPI со стороны мастера для Artery

	Имя файла	Назначение файла
21	ArterySysTickClock.cpp	Драйвер часов для Artery
22	ArterySysTickClock.h	Драйвер часов для Artery
23	ArterySysTickTimerDevice.cpp	Драйвер таймера SysTick для Artery
24	ArterySysTickTimerDevice.h	Драйвер таймера SysTick для Artery
25	ArteryTimerDevice.cpp	Драйвер таймера для Artery
26	ArteryTimerDevice.h	Драйвер таймера для Artery
27	ArteryUartDevice.cpp	Драйвер UART для Artery
28	ArteryUartDevice.h	Драйвер UART для Artery
29	ArteryWdtDevice.cpp	Драйвер WDT для Artery
30	ArteryWdtDevice.h	Драйвер WDT для Artery
31	DisableInterrupts.h	Реализация DisableInterrupts под Artery
32	GpioInit.h	Инициализация GPIO для Artery
33	CAN1_SE_InterruptDelegate.h	Делегат, вызываемый по прерыванию CAN1_SE
34	I2C2_ERR_InterruptDelegate.h	Делегат, вызываемый по прерыванию I2C2_ERR
35	I2C2_EVT_InterruptDelegate.h	Делегат, вызываемый по прерыванию I2C2_EVT
36	SysTick_InterruptDelegate.h	Делегат, вызываемый по прерыванию SysTick
37	TMR1_OVF_TMR10_InterruptDelegate.h	Делегат, вызываемый по прерыванию TMR1_OVF_TMR10
38	TMR2_GLOBAL_InterruptDelegate.h	Делегат, вызываемый по прерыванию TMR2_GLOBAL
39	TMR3_GLOBAL_InterruptDelegate.h	Делегат, вызываемый по прерыванию TMR3_GLOBAL
40	TMR4_GLOBAL_InterruptDelegate.h	Делегат, вызываемый по прерыванию TMR4_GLOBAL
41	USART1_InterruptDelegate.h	Делегат, вызываемый по прерыванию USART1
42	USART3_InterruptDelegate.h	Делегат, вызываемый по прерыванию USART3
43	USBFS_H_CAN1_TX_InterruptDelegate.h	Делегат, вызываемый по прерыванию USBFS_H_CAN1_TX
44	USBFS_L_CAN1_RX0_InterruptDelegate.h	Делегат, вызываемый по прерыванию USBFS_L_CAN1_RX0

	Имя файла	Назначение файла
45	Heap.cpp	Простая реализация кучи без удаления для микроконтроллеров
46	Atomic.h	Интерфейс атомарных операций (не могут быть прерваны другим потоком или прерыванием)
47	CanErrorFlags.h	Структура, содержащая флаги ошибок CAN
48	CanMessage.h	Структура, хранящая сообщение CAN
49	Delegate.h	Делегаты, позволяют ссылаться на функцию или метод класса в едином виде
50	DelegateBase.h	Внутренняя часть делегатов
51	InterruptDelegate.h	Тип делегата, который вызывается для обработки прерывания
52	ICanDevice.h	Интерфейс драйвера CAN
53	IClock.h	Часы, показывающие текущее время
54	IEnableDisableDevice.h	Устройство, поддерживающее включение/выключение
55	IGpioInput.h	Интерфейс драйвера GPIO, сконфигурированного на вход
56	IGpioInterrupt.h	Интерфейс драйвера External interrupt/Event controller
57	IGpioOutput.h	Интерфейс драйвера GPIO, сконфигурированного на выход
58	II2cBus.h	Интерфейс драйвера шины I2C со стороны Мастера
59	IMemory.h	Интерфейс памяти с произвольным побайтовым доступом
60	INorMemory.h	Интерфейс NOR-памяти
61	IRebootService.h	Интерфейс для перезапуска процессора (программы)
62	ISensorChannelDeviceSide.h	Интерфейс канала измерения аналогового датчика (АЦП и т.п.) со стороны драйвера
63	ISensorDevice.h	Интерфейс драйвера аналогового датчика (АЦП и т.п.)
64	ISleepService.h	Интерфейс для реализации задержки на определённое время
65	ISpiBus.h	Интерфейс шины SPI со стороны мастера
66	ITimerDevice.h	Интерфейс драйвера таймера, реализованного в железе

Имя файла		Назначение файла
67	IUartDevice.h	Интерфейс драйвера UART
68	IWatchdogDevice.h	Интерфейс драйвера сторожевого таймера
69	OperationStatus.h	Статус выполнения операции
70	ITimer.h	Интерфейс таймера
71	TimePoint.h	Момент локального времени (метка времени)
72	TimeSpan.h	Промежуток времени, конвертирующийся в разные понятные единицы
73	Uint.h	Возвращает тип, вмещающий N бит
74	AlsnToSaut485Sender.cpp	Подставляет в страницу БС-КЛУБ данные KptDecoder и IAlsnProvider
75	AlsnToSaut485Sender.h	Подставляет в страницу БС-КЛУБ данные KptDecoder и IAlsnProvider
76	CanAlsnProvider.cpp	Предоставляет данные об изменении состояния огибающей сигнала АЛСН по CAN-сообщениями МП-АЛС
77	CanAlsnProvider.h	Предоставляет данные об изменении состояния огибающей сигнала АЛСН по CAN-сообщениями МП-АЛС
78	IAlsnProvider.h	Интерфейс для класса, предоставляющего данные об изменении состояния огибающей сигнала АЛСН
79	KptDecoder.cpp	Расшифровывает посылки КПТ под данным IALsnProvider
80	KptDecoder.h	Расшифровывает посылки КПТ под данным IALsnProvider
81	BsClubFacade.cpp	Фасад модуля БС-КЛУБ
82	BsClubFacade.h	Фасад модуля БС-КЛУБ
83	CanToSaut485Gateway.cpp	Передаёт данные о подсистеме КЛУБ из CAN в линию RS-485 для работы подсистемы САУТ
84	CanToSaut485Gateway.h	Передаёт данные о подсистеме КЛУБ из CAN в линию RS-485 для работы подсистемы САУТ
85	BsDpsCanLogMessages.h	Коды лог-сообщений модуля БС-ДПС и МПХ
86	BsDpsMain.cpp	Главный файл алгоритмов БС-ДПС
87	BsDpsMain.h	Главный файл алгоритмов БС-ДПС

	Имя файла	Назначение файла
88	DiameterCorrector.cpp	Корректировка диаметра бандажа
89	DiameterCorrector.h	Корректировка диаметра бандажа
90	MalfunctionSaver.cpp	Сохраняет неисправность ДПС в энергонезависимую память
91	MalfunctionSaver.h	Сохраняет неисправность ДПС в энергонезависимую память
92	MileageSaver.cpp	Сохраняет пробег в энергонезависимую память
93	MileageSaver.h	Сохраняет пробег в энергонезависимую память
94	BothUnpluggedDetector.h	Контролирует обрыв двух ДПС
95	ConfidentGoodness.h	Рассчитывает «достоверную исправность»
96	DeviationSupervisor.h	Интегральная характеристика, показывающая разнятся ли показания скоростей
97	DpsChooser.cpp	Выбирает активный ДПС
98	DpsChooser.h	Выбирает активный ДПС
99	EcAdjust.cpp	Подстройка под электронную карту
100	EcAdjust.h	Подстройка под электронную карту
101	IpdCommunications.cpp	Коммуникации модуля ИПД по линиям связи
102	IpdCommunications.h	Коммуникации модуля ИПД по линиям связи
103	NeighbourPhaseSynchronizer.cpp	Корректирует период измерения, чтобы оба полукомплекта завершали измерение одновременно
104	NeighbourPhaseSynchronizer.h	Корректирует период измерения, чтобы оба полукомплекта завершали измерение одновременно
105	NeutralInsertion.cpp	Рассчитывает расстояние до нейтральной вставки
106	NeutralInsertion.h	Рассчитывает расстояние до нейтральной вставки
107	Saut485IpdPage.h	Страница для САУТ-485 для взаимодействия с модулем ИПД
108	CanSpeedEmulator.cpp	Обработчик команд эмуляции скорости по линии CAN

	Имя файла	Назначение файла
109	CanSpeedEmulator.h	Обработчик команд эмуляции скорости по линии CAN
110	DpsEmulationParameters.h	Параметры для эмуляции движения ДПС
111	LEmulationParametersHelper.cpp	Рассчитывает параметры для эмуляции
112	EmulationParametersHelper.h	Рассчитывает параметры для эмуляции
113	Saut485SpeedEmulator.cpp	Обработчик команд эмуляции скорости по линии RS-485
114	Saut485SpeedEmulator.h	Обработчик команд эмуляции скорости по линии RS-485
115	SpeedEmulator.cpp	Эмулятор движения
116	SpeedEmulator.h	Эмулятор движения
117	IpdFacade.cpp	Фасад всего, что связано с ИПД
118	IpdFacade.h	Фасад всего, что связано с ИПД
119	IpdModule.cpp	Модуль измерения пути и скорости
120	IpdModule.h	Модуль измерения пути и скорости
121	DirectionCalculator.h	Вычисляет направление движения по направлению вращения, позиции и прочему
122	DpsStateProcessor.h	Измеряет параметры движения, обрабатывая последовательность состояний порта ДПС
123	PreciseDistance.h	Расстояние, хранимое с максимальной точностью и удобно конвертирующееся в дециметры
124	SpeedMeasurer.h	Рассчитывает скорость
125	MphForIpdProvider.cpp	Берёт из МПХ параметры, нужные для работы ИПД
126	MphForIpdProvider.h	Берёт из МПХ параметры, нужные для работы ИПД
127	DpsParameters.h	Параметры ДПС
128	DpsPosition.h	Позиция установки ДПС
129	DpsRotation.h	Направление вращения ДПС
130	MovementDirection.h	Направление движения
131	SystemConfiguration.h	Параметры модуля ИПД, получаемые из конфигурации системы БЛОК (МПХ)

	Имя файла	Назначение файла
132	SystemState.h	Параметры модуля ИПД, описывающие текущее состояние системы БЛОК
133	AllCellsToCan.cpp	Выдаёт значения всех записанных ячеек МПХ в CAN
134	AllCellsToCan.h	Выдаёт значения всех записанных ячеек МПХ в CAN
135	CanPeriodicalTransmitter.cpp	Отправляет в CAN периодические сообщения МПХ
136	CanPeriodicalTransmitter.h	Отправляет в CAN периодические сообщения МПХ
137	CanRequestTransponder.cpp	Отвечает на запросы МПХ из CAN
138	CanRequestTransponder.h	Отвечает на запросы МПХ из CAN
139	OutOfConfigurationController.cpp	Выполняет перезагрузку, обнаружив что МПХ вышел из конфигурации ЦО
140	OutOfConfigurationController.h	Выполняет перезагрузку, обнаружив что МПХ вышел из конфигурации ЦО
141	Cell.h	Ячейка МПХ
142	CellIds.h	Список использованных ячеек МПХ
143	ICellManager.h	Интерфейс для работы с МПХ
144	InitialRequesterCellManagerDecorator.cpp	Опрашивает все ячейки при старте и уведомляет подписчиков ICellManager
145	InitialRequesterCellManagerDecorator.h	Опрашивает все ячейки при старте и уведомляет подписчиков ICellManager
146	Internals\Task.h	Задача в очереди МПХ
147	TaskSimplifyQueue.cpp	Очередь, которая удаляет дублирующиеся задачи
148	TaskSimplifyQueue.h	Очередь, которая удаляет дублирующиеся задачи
149	NiiasCellManagerDecorator.cpp	Подменяет номера ячеек в соответствии с алгоритмом НИИАС
150	NiiasCellManagerDecorator.h	Подменяет номера ячеек в соответствии с алгоритмом НИИАС
151	QueuedCellManager.cpp	Обеспечивает очередь при работе с МПХ
152	QueuedCellManager.h	Обеспечивает очередь при работе с МПХ
153	MemoryMap.h	Константы, определяющие размеры и размещение в памяти элементов МПХ
154	MphFacade.cpp	Базовый класс МПХ

	Имя файла	Назначение файла
155	MphFacade.h	Базовый класс МПХ
156	MphExchangeStateMachine.cpp	Синхронизация МПХ при старте
157	MphExchangeStateMachine.h	Синхронизация МПХ при старте
158	OverallChecksumCounter.cpp	Считает общую контрольную сумму по всем ячейкам МПХ
159	OverallChecksumCounter.h	Считает общую контрольную сумму по всем ячейкам МПХ
160	CellStoreWithSautCard.cpp	Декоратор для ICellStore, составляющий Карточку локомотива САУТ из ячеек МПХ
161	CellStoreWithSautCard.h	Декоратор для ICellStore, составляющий Карточку локомотива САУТ из ячеек МПХ
162	MphSautSynchronizer.cpp	Синхронизирует данные в Карточке локомотива САУТ и МПХ
163	MphSautSynchronizer.h	Синхронизирует данные в Карточке локомотива САУТ и МПХ
164	Saut485CardFlasher.cpp	Прошивка Карточки через режим программирования по САУТ-485
165	Saut485CardFlasher.h	Прошивка Карточки через режим программирования по САУТ-485
166	Saut485CardPage.h	Страница для САУТ-485 с данными Карточки Локомотива
167	SautCard.h	Карточка локомотива САУТ
168	SautCardInMemory.cpp	Сохраняет и загружает Карточку локомотива САУТ из памяти
169	SautCardInMemory.h	Сохраняет и загружает Карточку локомотива САУТ из памяти
170	ExternalMemoryCellStore.cpp	Определяет формат хранения ячейки в памяти
171	ExternalMemoryCellStore.h	Определяет формат хранения ячейки в памяти
172	ICellStore.h	Интерфейс хранилища для МПХ
173	ReservingCellStore.cpp	Обёртка над ICellStore, резервирующая данные
174	ReservingCellStore.h	Обёртка над ICellStore, резервирующая данные
175	WearLevelingCellStore.cpp	Обёртка над ICellStore, размазывающая избранные ячейки по нескольким, чтобы увеличить ресурс памяти

	Имя файла	Назначение файла
176	WearLevelingCellStore.h	Обёртка над ICellStore, размазывающая избранные ячейки по нескольким, чтобы увеличить ресурс памяти
177	WearLevelingMemoryCounter.cpp	Счётчик в энергонезависимой памяти, размазанный для увеличения её ресурса
178	WearLevelingMemoryCounter.h	Счётчик в энергонезависимой памяти, размазанный для увеличения её ресурса
179	BsDpsUnit.cpp	Устройство «БС-ДПС» на линии САУТ-485
180	BsDpsUnit.h	Устройство «БС-ДПС» на линии САУТ-485
181	TestModeCommunications.cpp	Взаимодействие по CAN для тестового режима
182	TestModeCommunications.h	Взаимодействие по CAN для тестового режима
183	TestModeMphMemory.cpp	Тестирование памяти МПХ
184	TestModeMphMemory.h	Тестирование памяти МПХ
185	TestModeRs485.cpp	Тестовый режим для проверки RS-485
186	TestModeRs485.h	Тестовый режим для проверки RS-485
187	main.cpp	Главный файл БС-ДПС на Artery
188	project.h	Настройка процессора
189	CanLogger.cpp	Логер, позволяющий отправить лог в CAN из любого места
190	CanLogger.h	Логер, позволяющий отправить лог в CAN из любого места
191	CanLogMessage.h	Лог-сообщение для передачи через CanLogger
192	SharedCanLogMessages.h	Коды лог-сообщений уровня Shared
193	BlokSilentModeActivator.h	Включает режим тишины у CanProvider по команде системы БЛОК
194	CanBaudrateCalculator.cpp	Рассчитывает настройки для контроллера CAN
195	CanBaudrateCalculator.h	Рассчитывает настройки для контроллера CAN
196	CanIdCollisionHack.h	В случае возникновения коллизии в CAN на время выключает драйвер, чтобы дать возможность отправить сообщение другому устройству
197	CanProvider.cpp	Подключает клиентские CanSocket'ы к CAN

	Имя файла	Назначение файла
198	CanProvider.h	Подключает клиентские CanSocket'ы к CAN
199	CanSocket.cpp	Сокеты CAN, предоставляют возможность нескольким клиентам работать с CAN независимо
200	CanSocket.h	Сокеты CAN, предоставляют возможность нескольким клиентам работать с CAN независимо
201	ModbusMasterRtu.cpp	Modbus-RTU в режиме Мастера
202	ModbusMasterRtu.h	Modbus-RTU в режиме Мастера
203	Saut485Provider.cpp	Работа по линии RS-485 по протоколу САУТ
204	Saut485Provider.h	Работа по линии RS-485 по протоколу САУТ
205	Saut485Socket.h	Сокеты линии САУТ RS-485, предоставляют возможность нескольким клиентам работать с линией
206	Array.h	Массив, с длиной и итератором
207	ArrayEnumerator.h	Массив, с длиной и итератором
208	Event.h	Позволяет нескольким клиентам подписаться на событие
209	IEnumerable.h	Интерфейс перечисляемого объекта
210	IEnumerator.h	Интерфейс перечислителя
211	ListBase.cpp	Типонезависимая часть реализации списка
212	ListBase.h	Типонезависимая часть реализации списка
213	List.h	Реализация односвязного списка без new
214	Property.h	Позволяет подписываться на изменение значения свойства
215	SpiAvrProgrammer.cpp	Прошивальщик AVR по SPI
216	SpiAvrProgrammer.h	Прошивальщик AVR по SPI
217	ButtonEvent.h	Event, срабатывающий на изменение состояния GpioInput
218	PollingGpioInterrupt.cpp	Реализует IGpioInterrupt, опрашивая по таймеру GpioInput
219	PollingGpioInterrupt.h	Реализует IGpioInterrupt, опрашивая по таймеру GpioInput
220	UartSplitter.cpp	Разделяет один IUartDevice для двух потребителей

Имя файла	Назначение файла
221 UartSplitter.h	Разделяет один IUartDevice для двух потребителей
222 AuxResourceIdentification.cpp	Передаёт идентификационную информацию через AUX_RESOURCE по запросу от SYS_DIAGNOSTICS и при старте
223 AuxResourceIdentification.h	Передаёт идентификационную информацию через AUX_RESOURCE по запросу от SYS_DIAGNOSTICS и при старте
224 FudpRebooter.cpp	По протоколу FUDP отвечает идентификацией и перезагружается для передачи управления загрузчику
225 FudpRebooter.h	По протоколу FUDP отвечает идентификацией и перезагружается для передачи управления загрузчику
226 Saut485IdPage.h	Страница для САУТ-485 с идентификационными данными
227 HalfSet.h	Тип полукомплекта
228 Identification.h	Структура идентификационных данных
229 IdProvider.h	Интерфейс поставщика идентификационных данных
230 CanLoaderIdProvider.cpp	Поставщик идентификационных данных из CAN-загрузчика
231 CanLoaderIdProvider.h	Поставщик идентификационных данных из CAN-загрузчика
232 Saut100HIdentification.h	Структура идентификационных данных САУТ, хранящаяся по адресу 100H
233 Saut100HIdProvider.cpp	Поставщик идентификационных данных из области САУТ по адресу 100H
234 Saut100HIdProvider.h	Поставщик идентификационных данных из области САУТ по адресу 100H
235 Unit.h	Модуль в системе БЛОК
236 I2cMemory.cpp	Реализует IMemory для памяти, подключенной по I2C
237 I2cMemory.h	Реализует IMemory для памяти, подключенной по I2C
238 SpiMemory.cpp	Реализует IMemory для памяти с постраничной записью, подключенной по SPI

	Имя файла	Назначение файла
239	SpiMemory.h	Реализует IMemory для памяти с постраничной записью, подключенной по SPI
240	SpiNorMemory.cpp	Драйвер NOR-памяти, подключенной по SPI
241	SpiNorMemory.h	Драйвер NOR-памяти, подключенной по SPI
242	CanPositionChangeSource.cpp	Предоставляет данные об изменении пройденного пути из линии CAN
243	CanPositionChangeSource.h	Предоставляет данные об изменении пройденного пути из линии CAN
244	Distance.h	Расстояние между двумя позициями
245	IPositionChangeSource.h	Интерфейс источника изменения позиции (пройденного пути)
246	Position.h	Позиция (координата X). Разница двух позиций - пройденный путь.
247	PositionProvider.cpp	Предоставляет позицию (координату) для запрошенного момента времени в недавнем прошлом
248	PositionProvider.h	Предоставляет позицию (координату) для запрошенного момента времени в недавнем прошлом
249	Saut485PositionChangeSource.cpp	Предоставляет данные об изменении пройденного пути из линии RS-485
250	Saut485PositionChangeSource.h	Предоставляет данные об изменении пройденного пути из линии RS-485
251	ArrayQueue.h	Реализует очередь на памяти, выделенной во вне
252	FifoBuffer.cpp	FIFO-буфер
253	FifoBuffer.h	FIFO-буфер
254	TypelessArrayQueue.cpp	Типонезависимая часть очереди
255	TypelessArrayQueue.h	Типонезависимая часть очереди
256	IQueue.h	Интерфейс очереди
257	NorRingBuffer.cpp	Кольцевой буфер для NOR-памяти
258	NorRingBuffer.h	Кольцевой буфер для NOR-памяти
259	QueueProcessor.h	Автоматически достаёт элементы из очереди и выполняет над ними пользовательскую операцию

Имя файла	Назначение файла
260 NorVirtualMemory.cpp	Реализует интерфейс IVirtualMemory для работы с NOR-памятью
261 NorVirtualMemory.h	Реализует интерфейс IVirtualMemory для работы с NOR-памятью
262 VirtualFileSystem.cpp	Виртуальная файловая система, в которой имя папки определят память, а имя файла - адрес в этой памяти
263 VirtualFileSystem.h	Виртуальная файловая система, в которой имя папки определят память, а имя файла - адрес в этой памяти
264 CrcChecker.cpp	Проверяет контрольную сумму файлов прошивки
265 CrcChecker.h	Проверяет контрольную сумму файлов прошивки
266 FudpClient.cpp	Клиент протокола удалённого обновления прошивки FUDP
267 FudpClient.h	Клиент протокола удалённого обновления прошивки FUDP
268 IFileSystem.h	Интерфейс файловой системы для FUDP
269 IFirmwareSubmodule.h	Субмодуль протоколов удалённого обслуживания FUDP
270 IProgrammerStateProvider.h	Сообщает о том, подключен ли программатор
271 IPropertiesStorage.h	Интерфейс для доступа к свойствам FUDP
272 LoaderStateMachine.cpp	Машина состояний загрузчика
273 LoaderStateMachine.h	Машина состояний загрузчика
274 PredefinedPropertiesStorage.cpp	Выдаёт неизменяемые параметры загрузчика из структуры
275 PredefinedPropertiesStorage.h	Выдаёт неизменяемые параметры загрузчика из структуры
276 CanRemoteSessionInitializer.cpp	Инициатор сессии протоколов удалённого обновления FUDP/LDP по линии CAN
277 CanRemoteSessionInitializer.h	Инициатор сессии протоколов удалённого обновления FUDP/LDP по линии CAN
278 InitIdentification.h	Данные о модуле, необходимые для открытия сессии удалённого обслуживания
279 IRemoteSessionInitializer.h	Инициатор сессии протоколов удалённого обновления FUDP/LDP
280 Submodule.h	Субмодуль протоколов удалённого обслуживания FUDP/LDP

	Имя файла	Назначение файла
281	ILogCatalog.h	Интерфейс хранилища логов для LDP
282	LdpClient.cpp	Клиент (модуль) протокола удалённого скачивания логов LDP
283	LdpClient.h	Клиент (модуль) протокола удалённого скачивания логов LDP
284	LogSubmodule.h	Субмодуль протоколов удалённого обслуживания LDP
285	Scheduler.h	Планировщик, ставящий задачи в очередь на выполнение
286	Timer.cpp	Высокоуровневый Timer, работающий на Scheduler
287	Timer.h	Высокоуровневый Timer, работающий на Scheduler
288	AuxResourceSimultaneousBoot.cpp	SimultaneousBoot через CAN-сообщения AUX_RESOURCE
289	AuxResourceSimultaneousBoot.h	SimultaneousBoot через CAN-сообщения AUX_RESOURCE
290	SimultaneousBoot.cpp	Обеспечивает одновременную загрузку и перезагрузку полукомплектов
291	SimultaneousBoot.h	Обеспечивает одновременную загрузку и перезагрузку полукомплектов
292	ITextLoggerImplementation.h	Интерфейс для класса, реализующего логирование текстовых сообщений
293	TextLogger.cpp	Логер, позволяющий отправить текстовое сообщение в лог из любого места
294	TextLogger.h	Логер, позволяющий отправить текстовое сообщение в лог из любого места
295	CanDateProvider.cpp	Показывает последнюю дату, переданную по CAN
296	CanDateProvider.h	Показывает последнюю дату, переданную по CAN
297	CanTime.h	Метка времени, которой помечаются CAN-сообщения в системе БЛОК
298	CanTimeSynchronizer.cpp	Часы, подстраивающиеся под общесистемное время системы БЛОК
299	CanTimeSynchronizer.h	Часы, подстраивающиеся под общесистемное время системы БЛОК
300	Clock.cpp	Реализация IClock, увеличивающая время по прерыванию ITimerDevice
301	Clock.h	Реализация IClock, увеличивающая время по прерыванию ITimerDevice

Имя файла	Назначение файла
302 Date.h	Структура данных, содержащая дату
303 TimeStamped.h	Добавляет метку времени к значению переменной
304 IMessageTransportProtocol.h	Интерфейс протокола транспортного уровня, передающего данные, разделённые на сообщения
305 IsoTp.cpp	Протокол транспортного уровня ISO-TP
306 IsoTp.h	Протокол транспортного уровня ISO-TP
307 IsoTpReceiver.cpp	Приёмник протокола ISO-TP
308 IsoTpReceiver.h	Приёмник протокола ISO-TP
309 IsoTpTransmitter.cpp	Передатчик протокола ISO-TP
310 IsoTpTransmitter.h	Передатчик протокола ISO-TP
311 Bytes.h	Удобные штуки для перестановки байтов и битов
312 CrcCalculator.h	Считает CRC
313 Hysteresis.h	Округление с гистерезисом
314 Mathematics.h	Удобные штуки, связанные с математикой
315 Ranged.h	Число с заданным диапазоном
316 ThreadSafe.h	Включает atomic, если для копирования переменной требуется больше одной ассемблерной инструкции

3.2. Принцип работы датчика ДПС

3.1.1. Датчик ДПС устанавливается на ось колёсной пары и даёт возможность определить скорость и направление вращения этой оси.

С осью датчика соединяется зубчатое колесо, имеющее N одинаковых зубьев, и, соответственно, N прорезей между зубьями. Размеры прорезей считаются равными размеру зубьев. На корпусе датчика устанавливается две оптопары на одинаковом расстоянии от оси.

При вращении колесо последовательно перекрывает оптопары. Таким образом, каждая оптопара генерирует импульсы по своему каналу. По частоте импульсов с

любого канала можно определить скорость вращения. По тому, по какому каналу импульс приходит первым, можно определить направление вращения.

4. ИСПОЛЬЗУЕМЫЕ ТЕХНИЧЕСКИЕ СРЕДСТВА

4.1. Блок БС-ДПС работает в составе ПО БЛОК-М и иных локомотивных систем.

Используемый микроконтроллер – AT90CAN128 (ЦП – 16 МГц, ОЗУ – 4 Кб, ПЗУ – 128 Кб).

5. ВЫЗОВ И ЗАГРУЗКА

5.1. Программа начинает автоматически выполняться при подаче электропитания на комплекс БЛОК-М.

6. ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

6.1. Входные данные

6.1.1. Для вычисления пройденного пути, скорости, ускорения и направления движения необходимы сигналы с датчиков ДПС. Для выбора активного датчика необходима информация о наличии тяги, передаваемая модулем ЦО по CAN-линии.

Для определения положения изостыка необходимы длительности импульсов огибающей сигнала АЛСН, передаваемые модулем МП-АЛС по CAN-линии. Для корректной передачи расстояния от изостыка модулю МП необходим обмен служебной информацией, производимый по линии связи RS-485.

Для согласования с электронной картой необходим её пройденный путь, передаваемый по CAN-линии модулем ЭК-СНС.

Запись постоянных характеристик происходит путём получения по линии CAN определённых сообщений на запись с номером ячейки и записываемыми данными. Чтение характеристик происходит путём выдачи сообщения, передаваемого по приходу сообщения-запроса.

Данные для передачи из CAN в RS-485 берутся из сообщений, передаваемых модулями ЦО и ЭК-СНС.

Для обеспечения ускоренной записи ЭК отправка всех сообщений в CAN-линию может быть отключена командой ЦО в сообщении MCO_OUT.

6.2. Выходные данные

6.2.1. Рассчитанные скорость и ускорение, определённые направление движения и исправность датчиков передаются в обе линии связи. В RS-485 передаётся оригинальный пройденный путь, а в CAN – результат, полученный после корректировки пройденного пути под электронную карту.

Расстояние до изостыка передаётся модулю МП по линии связи RS-485.

Постоянные характеристики передаются по обеим линиям связи.

Также в линию связи RS-485 передаётся описанная выше часть информации из CAN-линии.

ПРИЛОЖЕНИЕ 1

(Справочное)

ПЕРЕЧЕНЬ ПРИНЯТЫХ СОКРАЩЕНИЙ

АЛСН – автоматическая локомотивная сигнализация непрерывного действия;

БЛОК-М – безопасный локомотивный объединенный комплекса масштабируемого;

БС-ДПС – блок связи с датчиком пути и скорости;

БС-КЛУБ – блок связи с комплексным локомотивным устройством безопасности;

ДПС – датчик пути и скорости;

ИПД – измеритель параметров движения;

КПТ – кодовый путевой трансмиттер;

МПХ – модуль постоянных характеристик;

ПО – программное обеспечение;

САУТ – система автоматического управления торможением;

ЭК – электронная карта.

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

Изм.	Номера листов (страниц)				Всего листов (страниц) в документе	Номер документа	Номер сопроводительного документа и дата	Подпись	Дата
	Изменен-ных	Заменен-ных	Новых	Аннулированных					